HFS: Secure your server

This article is about *security* for HFS' users.

Many people ask: *Am I safe running this server software?*
100% safety on the net is, of course, impossible to achieve, but HFS is inherently secure.

For example, there are no current known security bugs in the latest version. HFS is open source, so anyone is able to easily check for security flaws (and we have many expert users).

Although it was not designed to be extremely robust, HFS is very stable and has been used for months without a restart (see forum topic).

Contents

1 Revision for Beginner Readers
2 What You Need:
3 Secure your Windows
4 What about strange logs?
5 HTTPS and SSL
6 Setting up Stunnel
7 Setup SSL

Revision for Beginner Readers [edit]

```
EDITED BY:
date: 08/16/2010
email: hfshomeip@gmail.com
reason: this wiki has helped me, I read it a couple of times before it makes sense to me. So my attempt is to revise it to make it more clearer for beginners. I only reorganized and clarified on some of the points where I found it unclear or confusing for beginner readers as I was and leave majority of it content intact.
credit goes to: ~GeeS~ and rejetto, and others who asked questions and contributed >>>here in the forum)
```

a) ~GeeS~ made the first attempt step-by-step tutorial in implementing HTTPS with HFS)
b) ~GeeS~ later submitted a updated revision which can be found here)
c) rejetto (the admin) who wiki ~GeeS~ works in (a & b) above which you will be reading below)

I went through pages 1-5 out of 10pages >>>>here in the forum **at time of this writing. It helped me with the basic understand and got my setup to work.**

What You Need: [edit]

The tutorial was written and tested on HFS version 2.0, and also worked for 2.2f and 2.3 beta built 226 on my setup. You need to google for the beta version. `(date 08/16/2010)`

HFS File Sever 2.2f
Stunnel v4.33
Registry mechanic
http://www.slproweb.com/products/Win32OpenSSL.html OpenSSL v1.0] - choose Light version x86 would do

YOU MAY WANT TO DOWNLOAD THEM ALL NOW DIRECTLY FROM THESE LINKS HERE SO YOU DON'T HAVE TO WORRY ABOUT IT LATER REFERENCES IN YOUR READING

All the available versions above are as of `date 08/16/2010`. They are all freewares.

Secure your Windows [edit]

Before worrying about HFS, you should be concerned about your Windows system. To secure your computer, follow this advice:

- Keep Windows updated with *Windows Update*.
- Use a good and updated firewall.
- Use a good and updated antivirus.

If you comply with these rules, the risks are very low.

What about strange logs? [edit]

If you see scary requests in the log, don't worry. They always address known security problems of **other** server software, not HFS. At the moment I write there's no known real security problem.

HTTPS and SSL [edit]

HTTPS is not supported natively at the moment. In the meantime, you can try adding HTTPS support by using STunnel. You can read an article on our forum where this discussed in detail.

Easy & secure data sharing on Windows with HFS & Stunnel for free ... an essay

The opportunity:
Today's widespread cable- and ADSL internet access offers a permanent connection to the worldwide web for home users. External hosting to serve private web pages or to share files is not necessary anymore.
HFS is a free, opensource http-file/web server for the win32 platform (Windows), it's very easy to use and runs "right out-of-the box" for down- and uploading files as well as for serving web pages directly from a PC at home. It can even travel on a disk or pendrive to serve from any PC at hand and addressable from the Net (keyword: portforwarding) or within a network.

The risk:
All web servers (not only HFS), which use HTTP, have a common weakness:
HTTP-traffic is transmitted in plain text and every bit of data travelling between a web server and a client (browser) can be intercepted and read by everyone who is in the chain passing data to the final destination. Even encoded usernames and passwords, which should protect web servers against unauthorized access, are easy to reveal. Only encrypted traffic (HTTPS) between server and client can protect precious private data against sniffing.
By encrypting the traffic between a server and its clients, a sniffer still is able to see which client IP is exchanging data with a certain web server at a certain time, but it is practically impossible (as long as the sniffer does not have the randomly generated private key) to decrypt the transmitted data.
While most modern browsers can handle encrypted traffic, the HFS server only supports insecure HTTP.

The solution:
Stunnel - a free, opensource multiplatform SSL tunneling proxyprogram- "is designed to work as an SSL encryption wrapper between remote client and local (inetd-startable) or remote server. It can be used to add SSL functionality to commonly used inetd daemons like POP2, POP3, and IMAP servers without any changes in the programs' code. Stunnel uses OpenSSL or SSLeay libraries for cryptography ..."
This just means, that Stunnel will be used to accept the client requests and establish an encrypted (HTTPS) connection, while Stunnel and the HFS server are exchanging non-encrypted data (HTTP).
A typical configuration of a PC with an https-enabled HFS server:
- Stunnel accepts requests from any IP on port 443, the HTTPS default port.
- Stunnel connects to HFS on an arbitrary chosen free port (e.g. 44300).
- HFS accepts requests on the chosen port, in this example 44300.
- Direct requests from clients to HFS on port 44300 have been blocked, except from 127.0.0.1 (localhost), where Stunnel resides.
- The PC and drives where HFS, Stunnel and the data reside are secured against unauthorized access.

Setting up Stunnel [edit]

How-to setup Stunnel for a SSL-secured HFS server, create a privatekey and self-signed server certificate

1. Stunnel is available from http://stunnel.mirt.net as a precompiled binary for Windows: "stunnel-4.33-installer.exe" at the time of writing. Install it and you will end up with:
- stunnel.exe (= the Stunnel program)
- stunnel.html (= the Stunnel manual)
- stunnel.conf (= the Stunnel configuration file)
- zlib1.dll, libssl32.dll and libeay32.dll (= openssl library files)
- stunnel.pem (= the default privatekey/certificate file)

Note: to get rid of any registry entries made by the installer, save the above files and uninstall Stunnel. (I read that Stunnel doesn't inject registry entries, but to be sure just do it anyway)

2. Run "stunnel.exe" and open the log. Find the version of openssl used for compiling with stunnel: "1.0.0" at the time of writing.

If you install and execute "stunnel.exe" for the first time and have not modified the "stunnel.conf" file you will not see "stunnel.log", (which can be viewed with notepad) inside the Stunnel directory after you have run "stunnel.exe" in step 2.

Go to your taskbar to find Stunnel icon and double click it to view the log.
Example:

```
2010.08.16 13:50:38 LOG5[3588:4124]: Reading configuration from file stunnel.conf
2010.08.16 13:50:38 LOG5[3588:4124]: Configuration successful
2010.08.16 13:50:38 LOG5[3588:4124]: No limit detected for the number of clients
2010.08.16 13:50:38 LOG5[3588:4124]: stunnel 4.33 on x86-pc-mingw32-gnu with OpenSSL 1.0.0
29 Mar 2010
```

```
2010.08.16 13:50:38 LOG5[3588:4124]: Threading:WIN32 SSL:ENGINE Sockets:SELECT,IPv6
```

Extract this version of "openssl.exe" from "openssl.zip" or download it directly to your "stunnel" directory from http://www.openssl.org or http://stunnel.mirt.net (.../openssl/binary-0.9.8d-zdll/openssl.exe).
-OR- download v1.0.0a light at the beginning of this page and install "Win32_OpenSSL_Light-1_0_0a.exe", after the installation copy openssl.exe in <code>"C:\OpenSSL-Win32\bin\openssl.exe"

What I did was,I created a "MY SERVER" folder in a C:\ directory. make first sub folder as "hfs server" and copy hfs.exe into c:\MY SERVER\hfs server because hfs file server is a self executable. no installation. now create a second folder "stunnel" and copy all the file listed above after the installation over to c:\MY SERVER\stunnel

Setup SSL [edit]

3. Open a text editor (e.g. notepad) and copy/paste the following entries:

```
[ req ]
default_bits = 2048
encrypt_key = yes
distinguished_name = req_dn
x509_extensions = cert_type

[ req_dn ]
countryName = Country Name (2 letter code)
countryName_default = XX
stateOrProvinceName = State or Province Name (full name)
localityName = Locality Name (eg, city)
0.organizationName = Organization Name (eg, company)
organizationalUnitName = Organizational Unit Name (eg, section)
0.commonName = Common Name (FQDN of your server)

[ cert_type ]
nsCertType = server
```

Save this file as "pem.conf" in the "c:\MY SERVER\stunnel" directory.

4. Delete the "stunnel.pem", which contains a default server certificate and privatekey.
It is a bad idea to use the stunnel.pem file shipped with stunnel except for testing!
In order to produce pem-file with a unique secure private key / server certificate, open a text editor (e.g. notepad) and copy/paste the following entries:

```
openssl.exe req -new -x509 -days 3650 -nodes -config pem.conf -out stunnel.pem -keyout stunnel.pem
```

Save this file as "create_pem.bat" in the stunnel directory. Run "create_pem.bat",
answer the questions in the dialog and enter whatever you like.

Note: The Common Name (FQDN) is required and should be the hostname of the machine running stunnel e.g. www.myhomeserver.net.
If you can access the machine by more than one hostname some SSL clients will warn you that the certificate is being used on the wrong host, so it's best to have this match the hostname users will be accessing.

Each time you run "create_pem.bat", a new "stunnel.pem" file with a unique random private key and self assigned server certificate with 10 years validity will be created.
It is extremely important to keep this stunnel.pem file secret! It contains your private key for the encrypted traffic! Do not back-up, but create a new one if necessary.

"stunnel.pem" can generated alternatively online and eliminated HERE on Stunnel.org

5. Edit "stunnel.conf" with a text editor (... notepad), select and delete everything in "stunnel.conf" Copy and Paste all the code below:

```
; Lines preceded with a ";" are comments
; Empty lines are ignored
; For more options and details: see the manual (stunnel.html)
```

```
; File with certificate and private key
cert = stunnel.pem
key = stunnel.pem

; Log (1= minimal, 5=recommended, 7=all) and log file)
; Preceed with a ";" to disable logging
debug = 5
output = stunnel.log

; Some performance tuning
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1

; Data compression algorithm: zlib or rle
compression = zlib

; SSL bug options / NO SSL:v2 (SSLv3 and TLSv1 is enabled)
options = ALL
options = NO_SSLv2

; Service-level configuration
; Stunnel listens to port 443 (HTTPS) to any IP
; and connects to port 44300 (HFS) on localhost
[https]
accept = 0.0.0.0:443
connect = 127.0.0.1:44300
TIMEOUTclose = 0
```

Save the edited "stunnel.conf".

On my setup, I had accept port as 443 without 0.0.0.0 and connect port as 44300 withouth 127.0.0.1 to be able to get it to work.
[https]
accept = 443
connect = 44300

6. Stunnel is now configured to accept HTTPS requests from any IP on port 443 of your PC and connects with HTTP to port 44300 on the same PC (127.0.0.1).
Do not forget: Port 443 and 44300 on this PC have to be opened in a firewall and routers have to forward port 443 to your PC. Do not forward port 44300 on your router.

7. Start HFS (version 2.1d at the time of writing) to listen on port 44300.
In Menu/Limits/Bans…, enter "\127.0.0.1" without the quotation marks and check "Disconnect with no reply" in order to ban every IP except 127.0.0.1 to block direct http access to HFS with a "Host not found" message.
Within a "friendly" network you could consider to add e.g. "\192.168.*" to allow direct HTTP access to HFS from all machines in your network.

8. Test your configuration carefully. You might to want to change the debug mode to debug = 7 in the stunnel.conf file for more log details.

9. Additionally, you might want also to have an HTTP welcome page, which links to your HTTPS enabled pages and contains instructions for your visitors how to handle a self signed server certificate and the related error messages of some browsers with it: Run a second, independent instance of HFS on port 80, modify the template and link from there to your secure Stunnel-HFS server.

10. Optionally, you can put your data, Stunnel and HFS on a pendrive with e.g. ./Myserver/Stunnel/stunnel.exe…, ./Myserver/HFS443/hfs443.exe…, ./Myserver/HFS44300/hfs44300.exe…, ./Myserver/Filesystem/… (renamed the two "hfs.exe" for convenience).
Configure HFS to save on file and the registry will kept clean. No admin rights are required. Run stunnel.exe, hfs443.exe and hfs44300.exe on any PC in a network for secure data exchange from PC to PC.
Note: Works fine, even in "hostile" networks. You could even rename the executables to some "innocent" names like "excel.exe", "powerpnt.exe" or "winword.exe" to obfuscate the running processes. Windows will not complain running different processes with the same name. Use your imagination and be aware of the risks!

11. Visit also stunnel.mirt.net, www.stunnel.org , www.openssl.org and www.rejetto.com for further readings.

Some important notes:

Because Stunnel connects from 127.0.0.1 (localhost) to HFS, functions of HFS which deal with IP numbers will be influenced.

Logs will will only show one client:127.0.0.1 (Stunnel on localhost). Use the Stunnel logs additionally to find out the requesting IP addresses.

Limits (bans, speed, number of simultaneous downloads …) will have influence on one IP only: 127.0.0.1 (Stunnel on localhost). Keep that in mind!

The ~progress window will show the actual total transfers of all clients (again: 127.0.01) Nice new feature, but also a privacy risk: the filenames of your data will be revealed. Either disable the progress-template (any decent browser has already a comparable window) or protect the server with a password to inhibit the use of the ~progress command for unauthorized clients.

Do not link parts of your pages to external, insecure (HTTP) servers. It is annoying and makes your site less trustful.

Data transfer will be somewhat slower, due to the de- and encryption of the data, but evidently the bandwith of the connection will be in most cases the limiting factor.

Finally, never ever share your Stunnel or HFS directories and take special care to protect them against unauthorized access … and keep your private key private.

So, enjoy secure serving and exchanging data. The more you give, the more you get

Some small print:
Unfortunately in some countries encryption or the discussion of encryption is still considered illegal.
In some countries the free exchange of data is still considered illegal under certain conditions.
I'm not a lawyer; in case of doubt get professional advice.
These publication is based on my today's best knowledge and reflect only a small part of the plenty more options of Openssl, Stunnel and HFS. No guarantees or whatsoever.

~GeeS~

If you get "Error binding service [https] to 0.0.0.0:443; bind: Permission denied (WSAEACCES) (10013); Service [https] closed (FD=392)", then you might have a program listening on port 443 already. For example, Teamviewer. Run the following commands to see what programs is using port 443.

```
C:\Program Files (x86)\stunnel>Netstat -a -n -o | find "443"
```

```
  TCP    0.0.0.0:443            0.0.0.0:0              LISTENING       1176
```

```
C:\Program Files (x86)\stunnel>tasklist | find /i "1176"
```

```
  TeamViewer_Service.exe        1176 Services                   0     13,128 K
```

This page was last modified on 27 April 2012, at 05:50.

This page has been accessed 156,320 times.

Content is available under GNU Free Documentation License 1.2 unless otherwise noted.

Privacy policy About rejetto wiki Disclaimers